Exam: TDIU11

Operating Systems

2023-03-24 kl: 14-18

On-call (jour): Ahmed Rezine, (Tel. 1938).

- You may answer in either English or Swedish. You can use a dictionary between English and another language.
- Use the exam wrappers distributed at the exam room. Do not forget to fill each wrapper with your anonymous ID.
- Read the instructions and all the assignments before you begin answering.
- Be **precise and clearly motivate** all statements and reasoning. If in doubt about a question, write down your interpretation and assumptions.
- Write clearly. Unreadable text will be ignored!
- The exam is graded U, 3, 4, 5 (preliminary limits: 20pts, 30pts and 35pts).

Problem 1 (Processes and scheduling, 12pts)

A system state is described as follows:

- The system is idle with an empty ready queue.
- Process P1 is waiting for keyboard input. Its next CPU burst is 2 ticks.
- Process P2 is waiting for network data. Its next CPU burst is 7 ticks.
- Process P3 is waiting for disk data. Its next CPU burst is 5 ticks.

The following list of events will take place starting at time 0:

Tick	Event
0	Network data is delivered
1	Disk data is delivered
3	Keyboard input is delivered

Table 1. Events

Questions:

1. Assume preemptive shortest remaining time first is used for scheduling:
    a. Give a Gantt diagram showing which process is running from tick 0 to 14. (2pts).
    b. What is the waiting time of process P1. (1pts)
    c. What is the turnaround of process P2. (1pts)

2. Use non-preemptive shortest job first:
    a. Give a Gantt diagram showing which process is running from tick 0 to 14. (2pts).
    b. What is the waiting time of process P1. (1pts)
    c. What is the turnaround of process P2. (1pts)

3. Starvation is possible for both scheduling algorithms. Describe a single workload (i.e, processes, arrival times and burst times) that would result in starvation for both scheduling algorithms. (2pts).

4. How can you modify a priority-based scheduling algorithm such as the preemptive shortest remaining time first to eliminate starvation? (2pts).

Problem 2 (Filesystem, 12pt)

Assume a 1 TiB (i.e., 2^{40}bytes) hard drive with 512 bytes physical blocks. Assume the system uses a FAT table with 4 bytes per entry.

1. What is the difference between logical and physical blocks? (2pts)

2. Suppose logical blocks are 1024 bytes each (i.e., 2^{10}bytes).

    a. How large is the FAT table? (2pts)
    b. How large should the disk be to have a table of at most 128MiB (i.e., 2^{27}bytes) assuming logical blocks are still 1024 bytes and each FAT entry is 4 bytes? (2pts)

3. Suppose in the following a magnetic hard drive where:
    a. Logical blocks can be read or written individually.
    b. To modify a logical block, the block needs to be first read to memory where the modification is performed and then rewritten to the disk.
    c. File inodes are already in main memory and you do not need to rewrite them back to the disk.
    d. For indexed allocation, all index blocks are already in main memory, and you do not need to write them back to the disk.
    e. Free blocks are known (no need to further access the disk to find them).

    Assume a data logical block is to be added in a file between the 5^{th} and the 6^{th} logical blocks. The new logical block will need to be written to the disk with new content from memory. The file had 10 data logical blocks at the beginning of the addition operation (only in the disk, not yet in memory) and will now have 11 data logical blocks (that should be in the disk after the addition operation is performed).

    How many I/O operations to the disk (i.e., read and writes of data logical blocks) do you need if the filesystem uses:

    i) Linked allocation (but not FAT) (2pts).
    ii) Indexed allocation (2pts).
    iii) Contiguous allocation (2pts).

Problem 3 (Memory management, 12pts)

Assume an architecture with 32 bits addresses (both physical and logical addresses are 32 bits). The architecture can be configured to use page sizes of either 4KiB (2^{12}bytes) or 1 MiB (2^{20} bytes). Assume each page entry is 4 bytes.

1) Assume pages are 1MiB large.
    a. How much space (in bytes) would a page table take assuming one level paging? Would it make sense to have more levels? Explain. (2pts)
    b. What information is cached by the TLB (translation look-aside buffer)? Explain why TLB is used? (2pts)
    c. Suppose access to main memory takes 200 nanoseconds ($200 \times 10^{-9} seconds$¿. What should the TLB hit ratio be (given as a percentage between hits and total queries) in order for the time taken to access a virtual address to be 202 nanoseconds in average. You can neglect TLB access time. (3pts).

2) Assume pages are 4KiB large.
    a. How much space (in bytes) would a page table take assuming one level paging? (1pts)
    b. Explain why two levels would be enough. How large (in bytes) would a page table take at each level? (2pts)
    c. What is the smallest size taken by all the page tables if a process that only accesses one valid page from which it can read and write data (2pts).

Problem 4 (Protection and Security, 4pts)

1. Suppose you want to revoke access to an object. Explain how this can be achieved when using access lists and when using capability lists. Which one is simpler? (2pts).

2. Consider this snippet from “man strcpy”:

```
NAME
       strcpy, strncpy - copy a string
SYNOPSIS
       #include <string.h>
       char *strcpy(char *dest, const char *src);
       char *strncpy(char *dest, const char *src, size_t n);
DESCRIPTION
The strcpy() function copies the string pointed to by src, including the
terminating null byte ('\0'), to the buffer pointed to by dest.  The strings may
not overlap, and the destination string dest must be large enough to receive the
copy.  Beware of buffer overruns! (See BUGS.)

The strncpy() function is similar, except that at most n bytes of src are
copied.   Warning:  If there is no null byte among the first n bytes of src, the
string placed in dest will not be null-terminated.

If the length of src is less than n, strncpy() writes additional null bytes to
dest to ensure that a total of n bytes are written.
```

Describe a situation where it can be more dangerous to use **strcpy** instead of **strncpy** to copy user inputs (e.g., old passwords) in the source code of the **passwd** executable (allowing users to change their respective password). Explain why is it dangerous? (Aim for 3-6 sentences). (2pts)

```
$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 68208 Nov 29 12:53 /usr/bin/passwd
```